Age generalization of emotion recognition models

Nigel Bosch
Assistant Professor, School of Information Sciences (iSchool) and Department of Educational Psychology
NCSA Faculty affiliate

Work done with Jinlin Zheng (currently interning at Citrix)
Visual emotion recognition

- Active appearance models
- Human intelligence
Visual emotion recognition (2)

• Feature extraction

• Machine learning
Visual emotion recognition (3)

- Deep learning
Fairness in machine learning

• If the model in question could impact people, it is important to make ethical considerations, like:

  • Are models equally accurate for people from different races, ages, locations, or other demographic characteristics?
    • (Focus of this work)

  • Are model decisions transparent (interpretable)?

  • Do applications of models have egalitarian outcomes?
Age generalization in AffectNet

• AffectNet is an image database (like ImageNet), with pictures of emotional expressions collected from the web
  • Approximately 500K images manually labeled by crowd workers
  • Many images with no face or other issues leaving ~134K labeled images

• Age estimated automatically via pretrained model

• Split data into 40+ years old and < 40 years old groups
Generalization testing and simulated bias

All data

Younger

Older
Main research goals

• How do different deep learning architectures fair in age generalization testing?
  • Architectures vary widely (e.g., ResNet vs. Inception)
  • Previous work shows some architectures have surprising properties (e.g., style transfer with VGG)
• Does pre-training make a difference?
• What implications do results have?
Deep learning models

• VGG16

• DenseNet121

• MobileNet

• Xception

• Hopefully others soon!
Running on HAL

• To do this well, it is necessary to choose appropriate hyperparameters for the task

  • Initial experiments showed the one learning rate that worked well with VGG16 produce chance-level results for Xception

  • No straightforward way (that I know of so far) to do smart hyperparameter search methods (e.g., TPE)

  • Thus, introduced learning rate as a command-line parameter
Learning rate in array jobs

```bash
#SBATCH --output="vgg16.%A_%a.out"  # %j for non-array jobs, %A %a for arrays
#SBATCH --array=0-7

lr_grid=(.05 .01 .005 .001 .0005 .0001 .00005 .00001)
lr=${lr_grid[$SLURM_ARRAY_TASK_ID]}

echo "Job started at "`date`
echo "Job on node $SLURMD_NODENAME"
export HDF5_USE_FILE_LOCKING=FALSE
module purge
module load cuda
. `/opt/apps/anaconda3/etc/profile.d/conda.sh`
conda activate pai36

echo "Learning rate: $lr"
python ../train_affectnet.py 25 vgg16 young $lr --pretrain

echo "Job ended at "`date`"
```
Model selection

• It is also necessary to train the model for an appropriate amount of time, and decide which learning rate is best

• Approach:
  • Train each model for 50 epochs
  • Use a small validation set to decide whether to save the best model or not
Saving the best model (Keras callback)

```python
model.fit_generator(
    train_generator,
    class_weight=class_weights,
    steps_per_epoch=train_generator.n // train_generator.batch_size,
    epochs=argv.num_epochs,
    validation_data=valid_generator,
    validation_steps=valid_generator.n // valid_generator.batch_size,
    callbacks=[
        nn_utils.ResumeModelCheckpoint(model_fname, save_best_only=True, verbose=1),
    ]
)
Custom callback to enable resuming

• Default Keras ModelCheckpoint callback assumes all model fitting done in one run of the program

• For this research, it is useful to be able to train a model over multiple runs (to play nice with HAL time limits)

• Custom callback saves validation accuracy to a CSV that matches the output model filename, and resumes training if that CSV already exists
Example model resuming

- Model run with command-line parameters:
  - Model type: mobilenet
  - Dataset partition: young
  - Learning rate: 0.005
  - Pretraining

  - Resulting filename: mobilenet_young__005__-pretrain.h5

- Model resuming callback will look for:
  - mobilenet_young__005__-pretrain.h5.csv
Preliminary results/next steps

• Validation results only so far

• The best accuracies are just over 50%
  • Validation set is 8-way with equal sized classes; thus, chance level = 12.5%

• Next steps
  • Finish training more models
  • Statistical comparisons of accuracy
    • Logistic regression to predict correct/incorrect per instance from age
    • McNemar’s test of marginal homogeneity to look for secondary model biases
Thanks for listening!

pnb@Illinois.edu